Communication Patterns

Communication Architecture for Clusters (CAC'06)
Rhodes Island, Greece

Rolf Riesen
rolf@cs.sandia.gov

Sandia National Laboratories

April 25, 2006

Talk Overview

1. Goal
2. Hybrid Approach
3. Measurements
4. Comparison to Other Work
5. Future Work
6. Summary

Talk Overview
Goal
Approach
Measurements
Related Work
Future Work
Summary
Appendix

Section Outline

1 Goal

2 Hybrid Approach

3 Measurements

4 Comparison to Other Work

5 Future Work

6 Summary

Goal

Goal

- Simulate a supercomputer; e.g., Red Storm, using federated discreate event simulators
- With enough fidelity to make future purchase and design decisions concerning things like:
  - CPU choice
  - Memory size and speed
  - Network interface
  - Topology
  - Application behavior
  - Research directions
  - etc.
- Created initial prototype with promising attributes
- This talk/paper presents first results and describes simulator

Section Outline

1 Goal

2 Hybrid Approach

3 Measurements

4 Comparison to Other Work

5 Future Work

6 Summary

Hybrid Approach

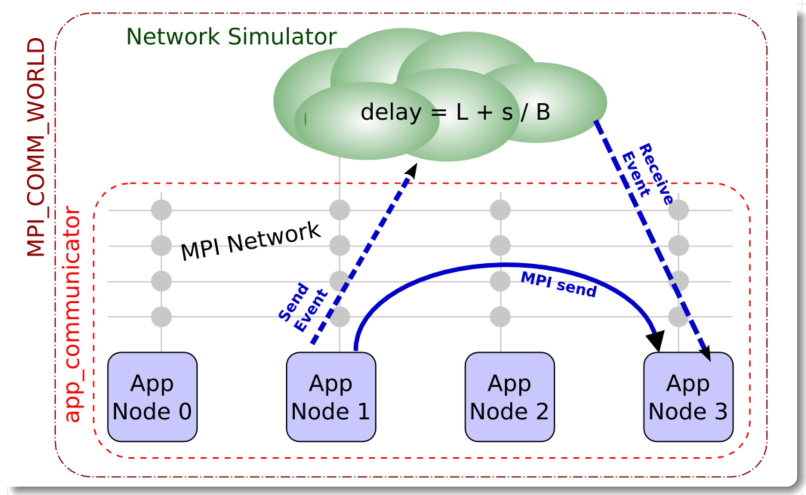
Hybrid simulator:

- App runs regularly and uses MPI to exchange data
- Each MPI send and receive generates an event to the network simulator
- Sim generates rcv events that are matched by clients
- Algorithm determines when and how to update virtual time on each node
- Use MPI wrappers and profiling interface
- Current network simulator uses simple model:

$$\Delta = \frac{s}{B} + L$$

Δ	network delay	B	network bandwidth
s	message size	L	network latency

Hybrid Approach



Hybrid Approach

```
int  MPI_Send(void *data,
              int  len,
              MPI_Datatype dt,
              int  dest,
              int  tag,
              MPI_Comm comm)
{
    t_x= get_vtime();

    //   Send the MPI message
    rc= PMPI_Send(data, len, dt, dest, tag,  comm);

    //   Send event to simulator
    event_send(t_x, len, dt, dest,  tag);

    return rc;
}
```

Hybrid Approach

```
int  MPI_Recv(void *data, int len, MPI_Datatype dt, int src,
               int  tag,  MPI_Comm comm, MPI_Status *stat)
{
     $t_1$= get_vtime();

     //  Receive the MPI message
     rc= PMPI_Recv(data, len, dt, src,  tag,  comm, stat);

     //  Wait for  the  matching event
     event_wait(&$t_x$, &$\Delta$, stat−>MPI_TAG, stat−>MPI_SOURCE);

     if  ($t_x + \Delta > t_1$)
          $t_3$= $t_x + \Delta$;
     else
          $t_3$= $t_1$;
     set_vtime($t_3$);  //  Adjust virtual   time
     return rc;
}
```

Hybrid Approach

- This approach seems to be new
- Combines low-intrusion measurement research with discrete event simulation
- Needs more validation, but seems to be very accurate
- Opens up many different and simple ways of evaluating applications and research directions

Section Outline

1 Goal

2 Hybrid Approach

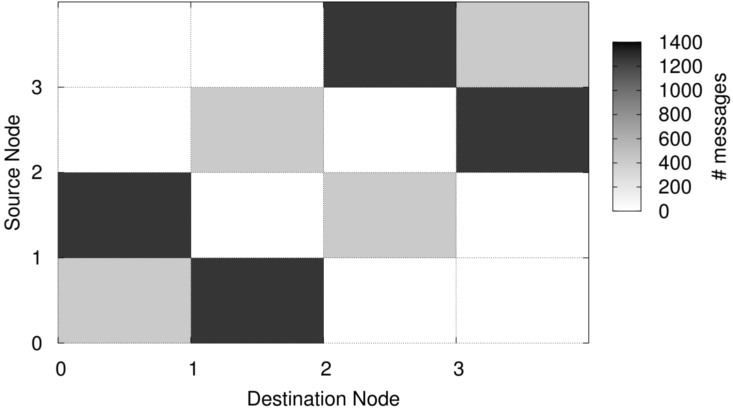
3 **Measurements**
- Message Density Distribution
- Data Density Distribution
- Collectives and Point-to-Point
- Types of Collectives
- Message Size Distribution

4 Comparison to Other Work

5 Future Work

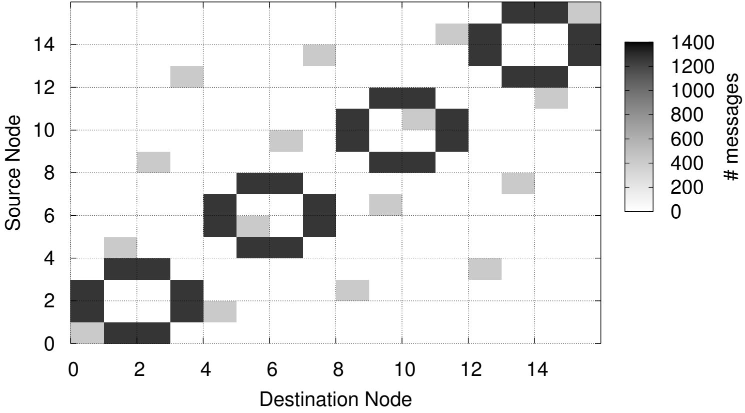
Message Density Distribution

CG (class A) message density distribution



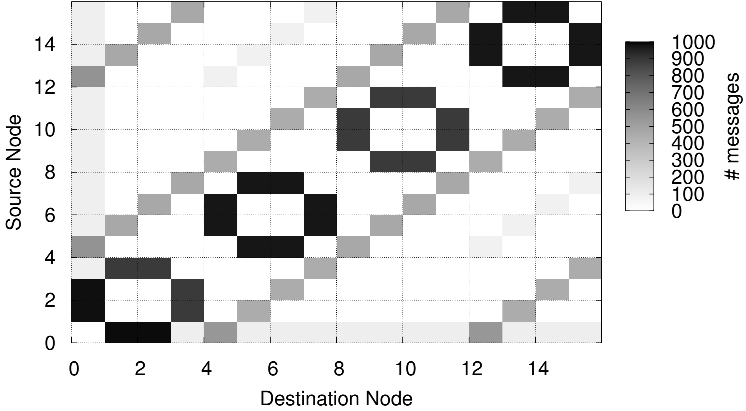
Message Density Distribution

CG (class A) message density distribution



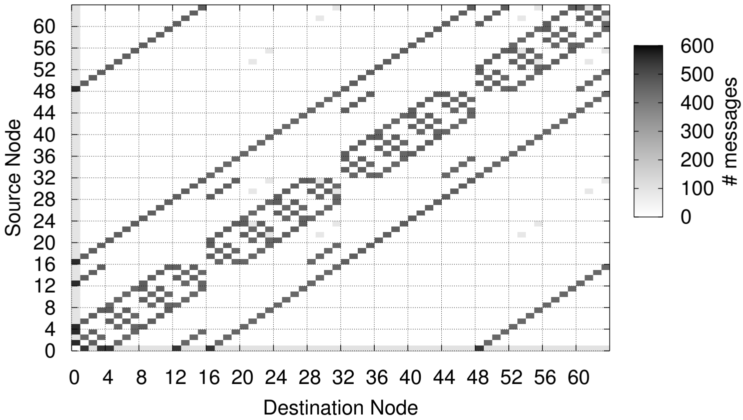
Message Density Distribution

MG (class B) message density distribution



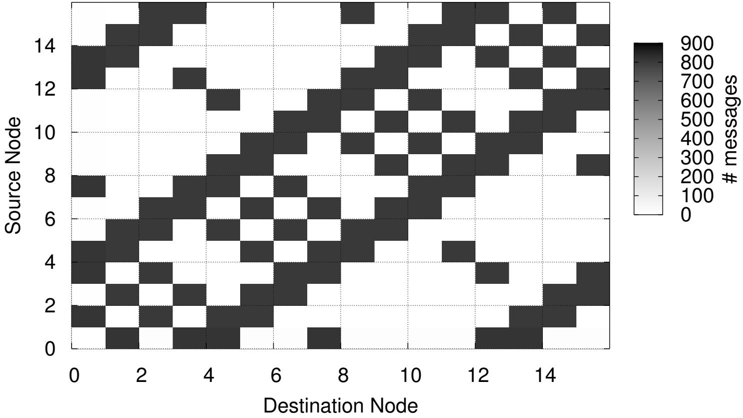
Message Density Distribution

MG (class B) message density distribution



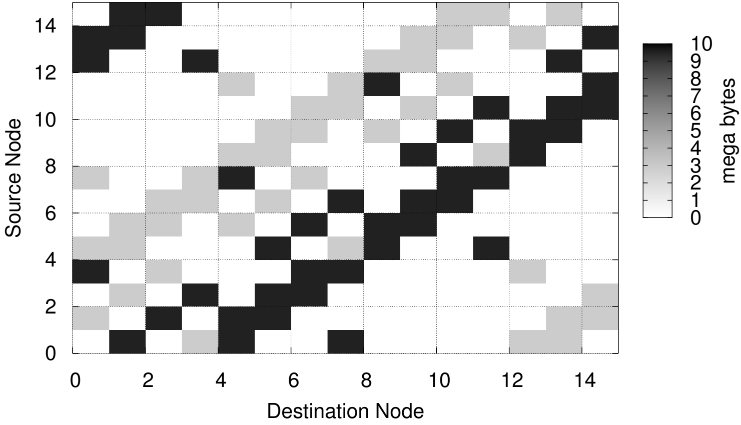
Data Density Distribution

BT (class A) message density distribution

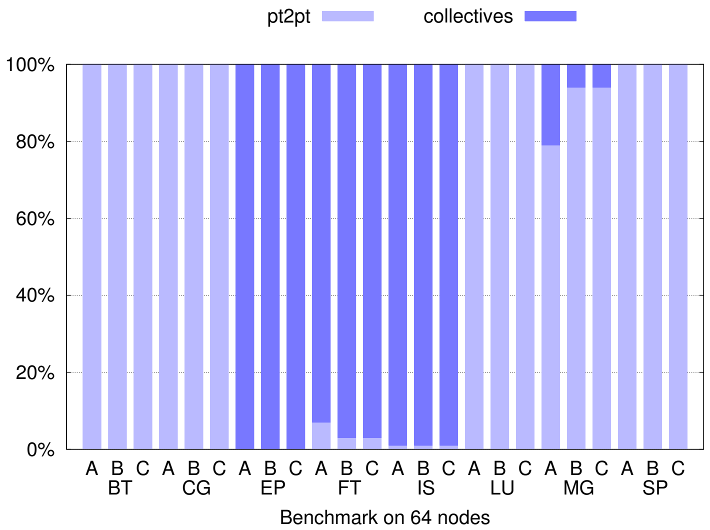


Data Density Distribution

BT (class A) data density distribution

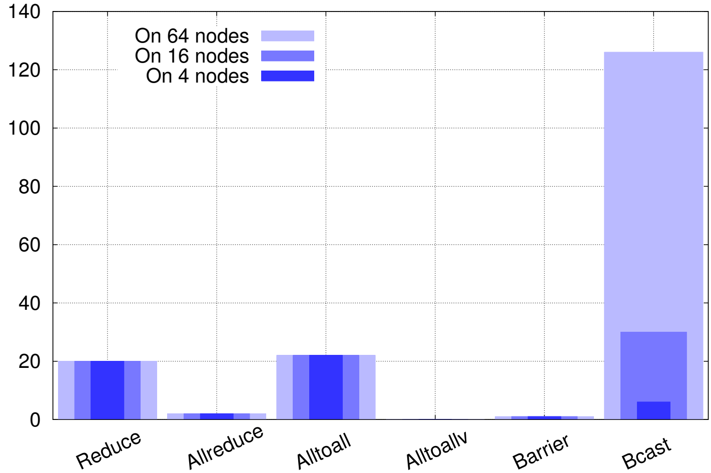


Collectives and Point-to-Point



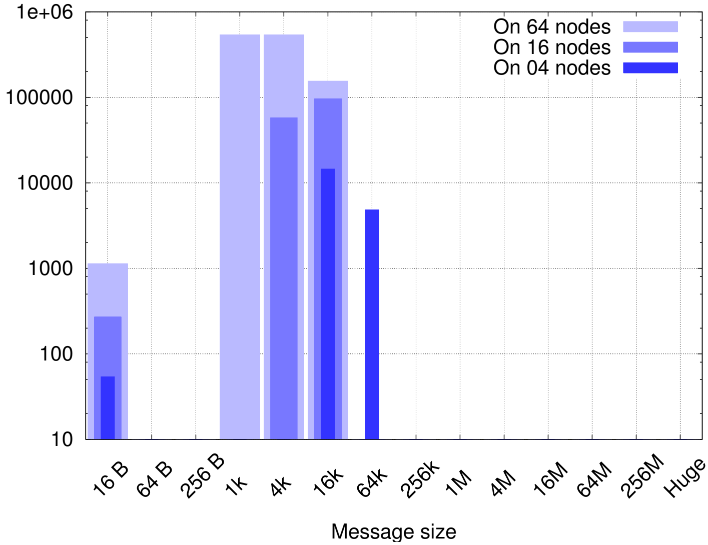
Types of Collectives

FT (class B) collectives



Message Size Distribution

Message sizes used by SP (class A)



Section Outline

1. Goal
2. Hybrid Approach
3. Measurements
4. Comparison to Other Work
5. Future Work
6. Summary

Comparison to Other Work

- No instrumentation code inserted into app
  - Rename **`main()`** (**`program`**) only change to app
- No disturbance of (virtual) runtime of app
  - Independent of amount of data collected.
- No extra memory needed on compute nodes to store trace data
- Language independent (e.g. Fortran and C for NAS)

Section Outline

1. Goal
2. Hybrid Approach
3. Measurements
4. Comparison to Other Work
5. Future Work
6. Summary

Future Work

Zero-Cost Collectives

- Putting collectives into NIC, building specialized NIC, or optimizing them is interesting
- How much application performance can be gained is not clear
- Simulator can assign $\Delta = 0$ to collectives and leave point-to-point alone

Future Work

Network Characteristics

- Simulator can change bandwidth and latency independently
- This can be used to evaluate application performance under varying network characteristics
- → predict impact of new network

Intrusion Free MPI Traces

- So far gathered only limited amounts of data
- Simulator can gather, and save to disk, large amount of data
    - Without changing application virtual time

Future Work

Network Characteristics

- Simulator can change bandwidth and latency independently
- This can be used to evaluate application performance under varying network characteristics
- → predict impact of new network

Intrusion Free MPI Traces

- So far gathered only limited amounts of data
- Simulator can gather, and save to disk, large amount of data
    - Without changing application virtual time

Talk Overview
Goal
Approach
Measurements
Related Work
Future Work
Summary
Appendix

Future Work

Continuing Work

- Need to incorporate more accurate network model
- This will allow simulation of congestion, and evaluation of topology choices, node allocation, etc.
- Move below MPI into NIC for more fine-grained simulation
- Incorporate non-network simulators; CPU and NIC sims

Section Outline

1 Goal

2 Hybrid Approach

3 Measurements

4 Comparison to Other Work

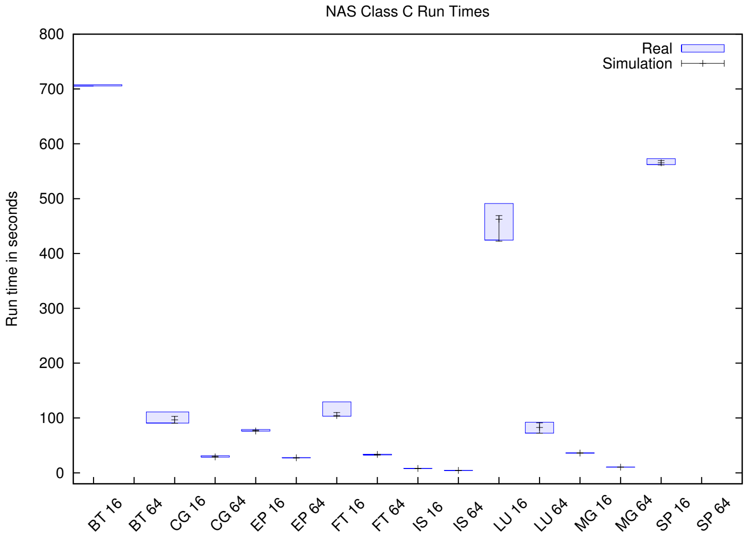
5 Future Work

6 **Summary**

Summary

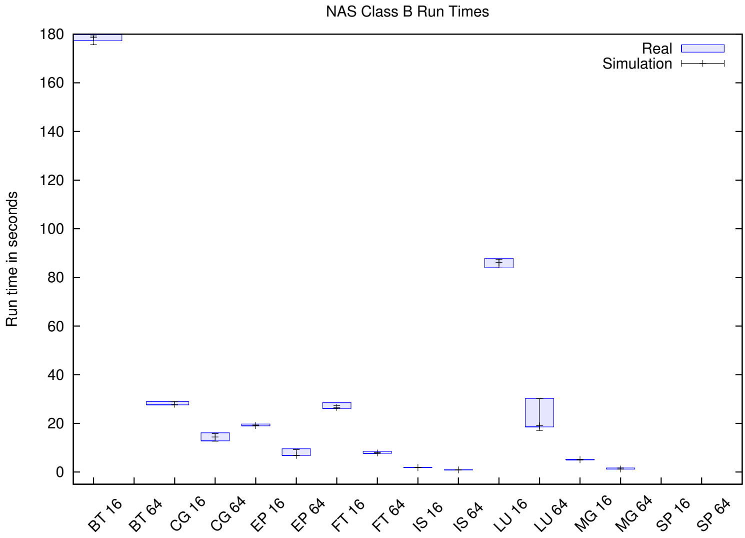
- Novel tool to collect MPI data
- Language independent
- Only linking with application needed
- Virtual runtime of application is not changed
- For this paper we collected data about
  - message density distribution
  - data density distribution
  - collectives versus point-to-point
  - number and type of collectives
  - message size distribution
- Lots of future possibilities

Validation

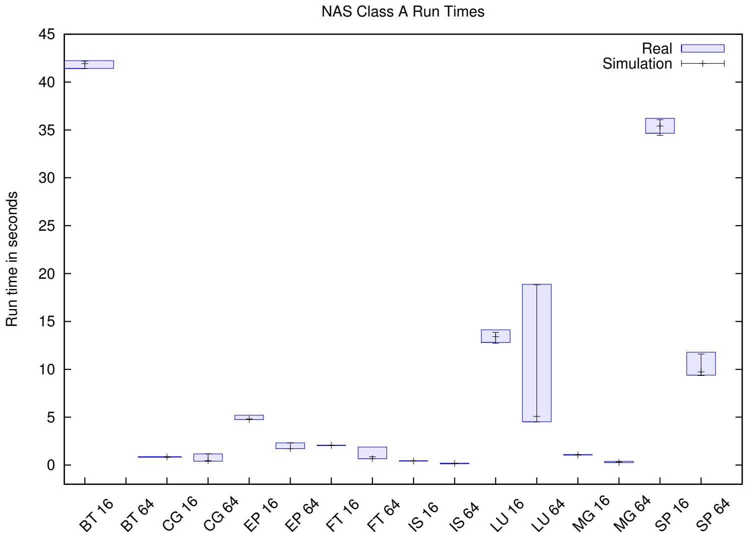


NAS Class C Run Times
Run time in seconds
0
100
200
300
400
500
600
700
800
Real
Simulation
BT 16
BT 64
CG 16
CG 64
EP 16
EP 64
FT 16
FT 64
IS 16
IS 64
LU 16
LU 64
MG 16
MG 64
SP 16
SP 64

Validation


NAS Class B Run Times
Run time in seconds
180
160
140
120
100
80
60
40
20
0
Real
Simulation
BT 16
BT 64
CG 16
CG 64
EP 16
EP 64
FT 16
FT 64
IS 16
IS 64
LU 16
LU 64
MG 16
MG 64
SP 16
SP 64

Validation


NAS Class A Run Times
Run time in seconds
0
5
10
15
20
25
30
35
40
45
Real
Simulation
BT 16
BT 64
CG 16
CG 64
EP 16
EP 64
FT 16
FT 64
IS 16
IS 64
LU 16
LU 64
MG 16
MG 64
SP 16
SP 64